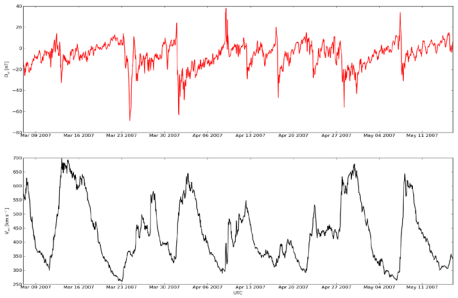
SpacePy In Action: Syntax and Output

spacepy commands in red || matplotlib commands in blue

Fetch and Plot OMNI Data

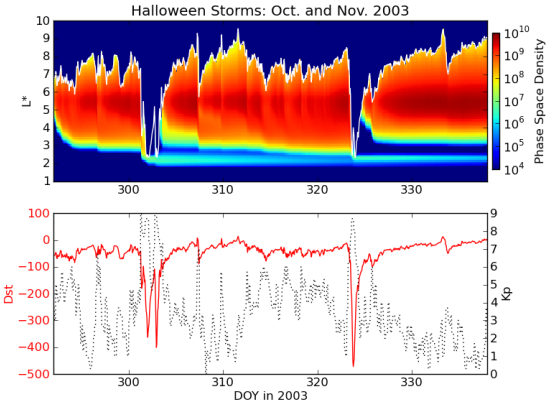
```
>>> import spacepy.omni as om
>>> import spacepy.time as spt
>>> ticks = spt.tickrange('2007-03-07T00:00:00', \
        '2007-05-16T00:00:00', 1./24.)
>>> data = om.get_omni(ticks)

>>> fig = figure()
>>> ax0 = fig.add_subplot(211)
>>> ax0.plot(data['UTC'], data['Dst'], 'r-')
>>> ax1 = fig.add_subplot(212)
>>> ax1.plot(data['UTC'],data['velo'], 'k-')
>>> ax0.set_ylabel('D$_{st}$ [nT]')
>>> ax1.set_ylabel('V$_{sw}$ [km s$^{1}$]')
>>> ax1.set_xlabel('UTC')
```



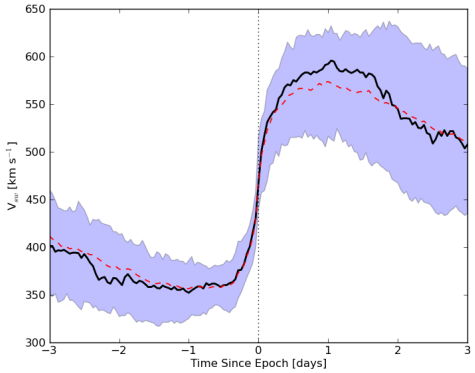
Perform Radiation Belt Simulation

```
>>> from spacepy import radbelt as rb
>>> import datetime as dt
>>> r = RBmodel()
>>> starttime = dt.datetime(2003,10,20)
>>> endtime   = dt.datetime(2003,12,5)
>>> delta = dt.timedelta(minutes=60)
>>> r.setup_ticks(startime, endtime, delta)
>>> r.evolve()
>>> r.plot(clims=[4,11])
```



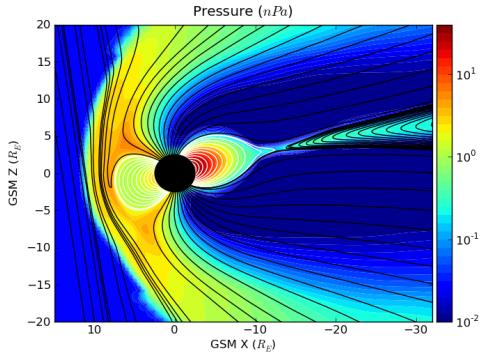
Superposed Epoch Analysis with OMNI

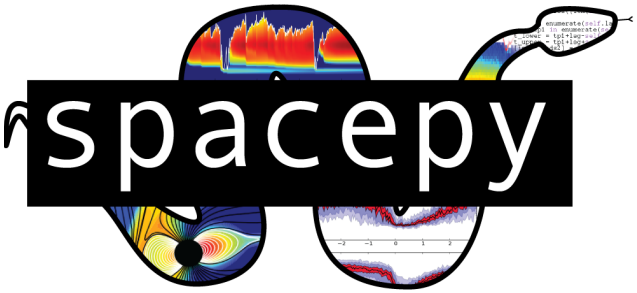
```
>>> from spacepy.seapy import *
>>> import spacepy.omni as om
>>> import spacepy.toolbox as tb
>>> epochs = readepochs('epochs.txt')
>>> st = datetime.datetime(2005,1,1)
>>> en = datetime.datetime(2009,1,1)
>>> einds, oinds = tb.tOverlap([st,en], \
        om.omnidata['UTC'])
>>> omni1hr = array(om.omnidata['UTC'])[oinds]
>>> delta = datetime.timedelta(hours=1)
>>> window= datetime.timedelta(days=3)
>>> sevx = se.Sea(om.omnidata['velo'][oinds], \
        omni1hr, epochs, window, delta)
>>> sevx.sea()
>>> sevx.plot()
```



Visualize 2D BATS-R-US Results

```
>>> import pybats.bats as bats
>>> obj = bats.Bats2d(`filename')
>>> obj.regrid(0.25, [40, 15], [30,30])
>>> fig = figure()
>>> ax = fig.add_subplot(111)
>>> obj.contourf(ax, `x', `y', `p')
>>> obj.add_body(ax)
>>> obj.add_planet_field(ax)
```





Python-Based Tools for the Space Science Community

- Quickly obtain data.
- Create publication quality plots.
- Perform complicated analysis easily.
- Run common empirical models.
- Change coordinates effortlessly.
- Harness the power of Python.

SpacePy Info
spacepy-info@lanl.gov

The SpacePy Team

Steve Morley
smorley@lanl.gov

Josef Koller
jkoller@lanl.gov

Dan Welling
dwelling@lanl.gov

Brian Larsen
balarsen@lanl.gov

Mike Henderson
mghenderson@lanl.gov

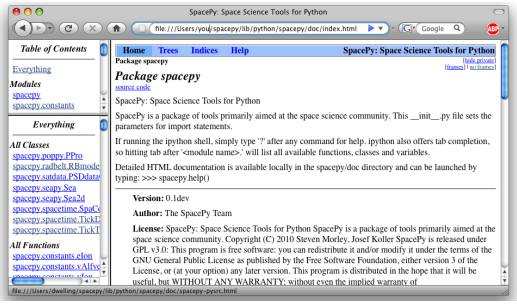
Los Alamos NATIONAL LABORATORY EST.1943

What is SpacePy?

SpacePy is a package of data analysis and visualization tools for the space science community. It is implemented in Python, a modern, object-oriented programming language. The goal of SpacePy is to allow users to perform as many research tasks as possible - from obtaining, analyzing and visualizing data to producing publication quality graphics - all inside a single, flexible programming language.

SpacePy functionality is broken into modules, which include:

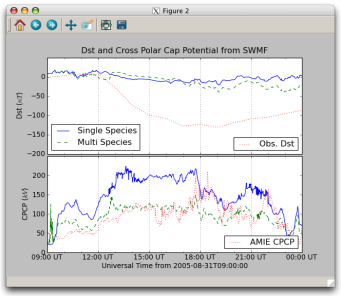
- **Time**, a module dedicated to properly handling the many complex time formats found in space science.
- **Omni**, a module for quickly obtaining data from the OMNI database.
- **Empiricals**, a library of commonly used empirical relationships.
- **SeaPy**, proper Superposed Epoch Analysis made easy.
- **OneraPy**, the powerful Onera library accessible through Python.
- **PyBats**, pure Python visualization for BATS-R-US and the SWMF.
- Association analysis, coordinate transformations, radiation belt modeling, CDF reading, and much more.



SpacePy documentation is available through your web browser or interactively as you work.

Why Python?

Python is a modern, object oriented language that has grown to be one of the most popular available because of its almost unlimited utility and power. Its **clear syntax**, **ease of use**, **portability**, and **rich standard library** have made it the first choice of many programmers around the world. Python is especially suited for scientific programming because of the **SciPy**, **NumPy**, and **MatPlotLib** packages. These libraries have become the foundation for SpacePy, creating a powerful numerical environment. Most importantly, ***all of this software is open source and free***.



MatPlotLib provides easy interactive plotting that can produce Postscript, PNG, PDF, and other high quality outputs.

What can I do with Python?

- Create GUIs to interactively explore your data and graphics.
- Interface web site forms with simple scripts.
- Document your code as it is written with versatile DocStrings.
- Wrap C, C++, Fortran, and other code to leverage the speed of compiled languages through a Python interface.
- Write CGI backends for web sites to bring your data and research directly to users.
- Customize your coding experience with advanced interactive shells and development environments.

Get Started

Python, Numpy, and MatPlotLib are required to use SpacePy. Linux users can find these in their package manager. Mac OSX users will find them in Fink or MacPorts. Distributions that include everything for most platforms can be found at:

http://enthought.com/products/getepd.php

Get SpacePy

Email spacepy-info@lanl.gov for information on how to get the latest version.

Get Results

Although it is brand new, SpacePy output has already been featured in three scientific articles. SpacePy is being leveraged for the Radiation Belt Storm Probe satellite mission science operations center and early mission planning. The user group and list of capabilities is constantly growing.